Justificación de la tecnología utilizada

INDICE – Justificación de la tecnología utilizada

4 Justificación de la tecnología utilizada

4.1 Introducción

Este Proyecto quiere confeccionar una herramienta para la construcción de revistas electrónicas de una forma genérica. Estando orientado a todos aquellos usuarios que desconozcan la utilización de tecnologías para la creación de su revista en Internet.

Por estos motivos incluimos las razones que nos han impulsado a utilizar una serie de herramientas y tecnologías para la confección de la aplicación

4.2 Arquitectura Física

Para explicar la arquitectura física se realizará una descripción sobre los componentes hardware que necesitará la aplicación.

4.2.1 Arquitectura Cliente/Servidor

Esta está formada por una parte por el cliente y otra por el servidor. De una forma sencilla se podría decir que la máquina cliente requiere un servido de la máquina servidor, pudiendo ser una máquina a la vez cliente y servidor.

La comunicación de los ordenadores conectados a Internet se rigen básicamente por esta arquitectura, permitiendo una gran flexibilidad en las comunicaciones.

El funcionamiento de esta arquitectura básicamente se fundamenta en la escucha por parte del servidor a través de los puertos de las distintas peticiones que van realizando los clientes. Las peticiones de los clientes también las realizan a través de puertos.

Una vez que el servidor recibe la petición se pondrá a atender al cliente, pero a la vez seguirá atendiendo a otras peticiones, ya que un servidor admite accesos simultáneos.

De entre todos los puertos que existen, el que más nos interesa para las comunicaciones en Internet es el 80, que se corresponde normalmente con el servicio HTTP (HyperText Transfer Protocol). Desde la parte cliente y con un navegador como son el Internet Explorer, Opera o Netscape nos conectaremos al servidor a través de este puerto obteniendo de ellos el servicio solicitado que no es otra cosa que las páginas Web.

4.2.2 Arquitectura de capas en el modelo Cliente / Servidor

Hay dos modelos posibles para la arquitectura de capas para el modelo Cliente/Servidor:

- Modelo de dos capas o Cliente-Servidor, Modelo1
- Modelo de n-capas, Modelo 2

Modelo de 2 capas, Modelo 1

Ésta es la arquitectura que ha sido más utilizada, y que consiste en una o más aplicaciones ejecutándose sobre máquinas cliente y conectadas a un servidor para poder trabajar. Las aplicaciones basadas en CGI y en servlets se basan en este modelo, en el que solamente hay dos capas: el cliente y el servidor.

Las dos capas, son:

- Nivel de Aplicación (Capa Cliente)
- Nivel de la Base de Datos (Capa de Servidor)

Las ventajas de este modelo son:

- La ventaja de este modelo es que resulta muy fácil de programar y el contenido dinámico de una página se genera rápidamente, en base a la petición del cliente y al estado de los recursos y además, a la hora de realizar modificaciones, solamente hay que abrir un archivo.

Las desventajas de este modelo son:

- Este modelo no es escalable, es decir, un gran número de peticiones simultáneas puede requerir el uso de muchos recursos del servidor, lo cual hará disminuir el rendimiento de la aplicación Web.
- El uso de este modelo también lleva acarreada una gran cantidad de código en las páginas, que aunque no representa un problema para los desarrolladores, si que se convierte en un problema para los diseñadores de páginas, los cuales no tienen porque tener conocimientos de programación.
- La aplicación y la base de datos tendrán que interactuar directamente, teniendo que modificar la aplicación si la base de datos cambia.

Modelo de n-capas, Modelo 2

En este caso se introducen una serie de capas intermedias que centralizan el acceso de la base de datos y se minimice el número de cambios en las aplicaciones en caso de que cambie la base de datos.

Ahora el cliente cuando envía una petición al servidor este la interpretará enviándolas a la nueva capa que accederá a la base de datos y devolverá la respuesta.

Se verá que ahora el modelo tendrá como mínimo tres capas:

- Nivel de aplicación cliente (Capa cliente)
- Nivel de aplicación servidor (Capa intermedia)
- Nivel de la Base de Datos (Capa de servidor)

Este modelo se puede dividir en grupos, uno correspondiente a las que manejan el flujo de la aplicación y la lógica que ello engloba, sin responsabilizarse de ningún tipo de presentación; es decir, este grupo representa el punto de entrada a la aplicación

Web. El otro nivel es el encargado de generar el código HTML que se enviará al cliente; es decir, solamente contendrán indicaciones de presentación y lógica para presentar contenido dinámico.

En definitiva, de lo que se trata en este modelo de arquitectura es de la creación de varias capas dentro del servidor, con una separación bien definida entre ellas, para poder desarrollar y posteriormente mantener de forma individual cada uno de los componentes que intervienen en la generación de la respuesta al cliente.

Este ahorro de esfuerzo ha conseguido que en la actualidad este modelo de n capas sea uno de los más utilizados en aplicaciones de Internet.

4.2.3 La Elección

Finalmente comentaremos que no hay una arquitectura ideal en el desarrollo de aplicaciones Web, sino que el uso de una u otra dependerá del tamaño y alcance de la aplicación, así como de la cantidad de recursos (humanos y monetarios) disponibles para llevar a cabo el desarrollo y del tiempo fijado para el desarrollo de esa aplicación.

En aplicaciones sencillas será suficientemente con la utilización del modelo Cliente Servidor. En cambio en aplicaciones relativamente simples que requieran accesos sencillos a base de datos, el modelo de tres capas puede ser la elección perfecta.

Si se trata ya de una gran aplicación, como es este el caso, será necesario pensar en una solución que permita realmente la escalabilidad, por lo que la solución de utilizar arquitectura de n-capas utilizando el patrón MVC (Ya comentada en el diseño) será la opción más adecuada.

Finalmente, decir que nuestra elección ha sido esta última (arquitectura de n-capas utilizando el patrón MVC). Al decantarnos por esta opción habrá que interrelacionar los componentes que intervienen en la generación de la respuesta en el servidor, es necesario que ya en el diseño de la aplicación se identifiquen claramente los objetos y sus interacciones, es decir, hay que modelar los objetos. Por este motivo ya se ha comentado con anterioridad el patrón MVC en el diseño.

4.3 Elección del Software

4.3.1 Introducción

Este es una parte fundamental para el desarrollo de la aplicación y por tanto vamos a exponer las distintas herramientas que se han seleccionado entre todas las alternativas.

4.3.2 Software del Cliente

Sistema Operativo

Gracias a la arquitectura cliente-servidor, el protocolo TCP/IP, el protocolo http y la gran flexibilidad proporcionada por los navegadores, no será necesario que el cliente tenga un S.O. determinado. Por lo tanto este aspecto se dejarán en manos del los

dueños o administradores de las máquinas clientes que quieran utilizar por Internet nuestra aplicación Web.

Navegador

Gracias al protocolo http y las especioficaciones HTML, JavaScript, Applets, hojas de estilo. La aplicación será compatible con cualquiera de los navegadores Web existentes que soporten las especificaciones mencionadas. No obstante habrá ciertas diferencias a la hora de visualizar las mismas páginas Web en los distintos navegadores, por tanto se optimizará para Internet Explorer ya que este es el más utilizado de los existentes en el mercado.

4.3.3 Software del Servidor

Sistema Operativo

No estaremos ligados a utilizar en la máquina servidor un sistema operativo en concreto ya que las peticiones de los clientes serán servidas por una aplicación que será un servidor Web que por lo general están disponibles en diversas plataformas.

No obstante hay dos grandes familias de sistemas operativos a la hora de motar servidores Web:

- Distribuciones GNU/Linux
- Familia Windows

Distribuciones GNU/Linux

Estos tienen una gran aceptación en el mundo de la informática a la hora de montar servidores Web, ya que presenta unos muy buenos resultados al mejor precio. Hay conocidas distribuciones gratuitas como son: Debian y Red Hat.

Sin embargo todos los usuarios coinciden en que la mayoría de las distribuciones exigen un alto nivel de conocimiento, tanto para instalar el sistema operativo como para configurarlo.

Familia Windows

Los sistemas operativos de Microsoft son los más difundidos, conocidos y utilizados en el mundo. Pero tienen serios problemas de estabilidad y numerosos bugs además se trata de un sistema operativo propietario y por tanto hay que pagar por utilizarlo. A cambio su instalación, configuración y manejo son muy fáciles e intuitivos lo que hace que siga teniendo el apoyo de muchos usuarios.

Servidor Web

A la hora de seleccionar nuestro servidor Web dependerá en gran medida del leguaje de programación elegido y del S.O. que se elija para motar el servidor Web.

Para comenzar explicaremos de un modo breve los dos servidores que copan el mercado;

- Apache

Es uno de los servidores más utilizados en Internet ya que se trata de un servidor muy potente, flexible, rápido, eficiente y que siempre está adaptado a nuevos protocolos http. Pero no queda hay la cosa, ya que se trata de un gran logro del software libre y por tanto se puede bajar gratuitamente desde Internet. Su fortaleza se debe a este hecho ya que se realimentación de las aportaciones que realiza los usuarios, al informar de fallos, al crear parches, al realizar ampliaciones, al aportan ideas, etc.

Otra de las grandes virtudes de Apache es que se encuentra el servidor Web disponible para varias plataformas, desde Debian, hasta Windows XP y se le puede incrustar nuevos módulos que le permitirán ejecutar código Script como son JSP, PHP, etc. La única pega es que la integración del apache con estos módulos suele ser muy complicada.

- Internet Information Server

Este no se trata de software libre como en el caso anterior, si no que es propiedad de Microsoft y por tanto hay que pagar por su uso. Incluye los servicios de : http,. https, ftp, smtp, nntp. Además es capaza de ejecutar varios motores Script como son: PHP, Cold Fusion, ASP, etc. Además viene integrado con el propio sistema operativo Windows XP, 2000 y NT. Un punto muy fuerte de este servidor Web es que es muy fácil instalarlo, activarlo y configurarlo.

Leguaje de Programación

Realizaremos un repaso entre las distintas opciones que tenemos en la actualidad para poder elegir el leguaje que mejor se adapte a nuestras necesidades.

Nuestra aplicación pretende al usuario crear y mantener revistas electrónicas que pueda interactuar con los distintos usuarios. Esto no se consigue con páginas Web estáticas, ya que sólo se podría ver los contenidos previamente generados, en cambio nuestra aplicación deberá generar contenidos dinámicos, por este motivo emplearemos aquellas herramientas que nos permitan el desarrollo de estas páginas Web dinámicas, y que a continuación veremos.

Common Gateway interface (CGI)

La primera forma de creación de contenido dinámico en páginas Web fue a través del mecanismo Common Gateway Interface (CGI), a través del cual los servidores Web pueden pasar información a páginas externas, que serán ejecutadas en el servidor Web para generar respuestas en tiempo de ejecución. El lenguaje Perl es el más utilizado para escribir este tipo de programas, aunque se puede utilizar cualquier lenguaje que genere programas que puedan ser invocados por el servidor Web, por ejemplo, cualquier lenguaje script soportado por el sistema operativo en donde esté corriendo el servidor Web, o un programa escrito en C y compilado, o una aplicación Java.`

La tecnología CGI no está exenta de ineficiencias que la hacen desaconsejable en aplicaciones medianamente complejas. Su inconveniente más importante deriva de su propia filosofía, la ejecución de programas externos para la generación de la respuesta al cliente. Cada petición genera un nuevo proceso externo, lo cual, en

servidores que atienden a muchas peticiones simultáneas, es una sobrecarga difícil de soportar.

FASTCGI

Existe otra opción que es el FastCGI. Es una alternativa al CGI estándar, cuya diferencia radica principalmente en el hecho de que el servidor crea un único proceso persistente por cada programa FastCGI en lugar de por cada solicitud del cliente. Aunque FastCGI es un paso en la dirección correcta, sigue teniendo problemas con la proliferación de procesos, ya que en todo momento existe al menos un proceso activo por cada programa FastCGI. Para manejar solicitudes concurrentes, habría que mantener un estanque de procesos, uno por cada solicitud. Considerando que cada uno de estos procesos puede estar ejecutando el intérprete de Perl, este modelo no parece tan distinto del CGI estándar. Una solución que FastCGI ofrece para este problema es su habilidad para distribuir todos estos procesos entre múltiples servidores.

ColdFusion

La tecnología ColdFusion, creada por Allaire, se basa en una serie de etiquetas HTML que soportan gran variedad de acciones para la generación de contenido dinámico. Estas etiquetas permiten, por ejemplo, realizar consultas a bases de datos, y mantienen una consistencia única con las etiquetas HTMl del resto de la página.

JavaScript

Es un lenguaje de script, interpretado, orientado a objeto, que permite introducir interactividad en documentos HTML y tiene la ventaja de que no necesita ser transmitido hacia el servidor, verificado y devuelto. Éste es ejecutado en el navegador del usuario.

Los scripts de Javascript pueden ser introducidos dentro de sus páginas de HTML. Con Javascript se puede dar respuesta a eventos iniciados por el usuario, eventos tales como la entrada de una forma o algún enlace. Las entradas son verificadas por la aplicación cliente y pueden ser transmitidas después de esto. Permite efectuar cálculos, efectos especiales, verificar formas, crear juegos, personalizar la gráfica, crear password de seguridad, y mucho más. Realiza el manejo de muchos recursos Web.

Server-Side JavaScript

Esta tecnología permite utilizar javaScript ejecutándose en el servidor web. La tecnología Server-Side JavaScript (SSJS) incorpora otras características al leguaje, como es el soporte para correo electrónico, control de sesiones e interoperabilidad con Java ejecutándose en el servidor a través de la tecnología LiveWire de Netscape.

El inconveniente de esta tecnología reside en su incompatibilidad, es decir, que aunque puede ejecutarse en cualquier plataforma o sistema operativo, al igual que Java, es una tecnología desarrollada específicamente para los servidores Netscape, o iPlanet.

PHP

Es una tecnología de código abierto que está actualmente en pleno crecimiento. PHP es el acrónimo de *Personal Home Page* y utiliza una sintaxis semejante al lenguaje C. Proporciona soporte para acceso a base de datos y dispone de extensiones para comunicarse con otros recursos.

PHP no es un producto comercial, sino que es la contribución realizada libremente por programadores a la comunidad. La consecuencia más importante de la naturaleza de código libre es que PHP está disponible para muchas plataformas, es compatible con Windows y con sistema Unix, y se puede utilizar en gran número de servidores web, como Apache, Microsoft IIS y iPlanet Enterprise Sever.

Active Server Pages (ASP)

Las ASP está basada en la inclusión de etiquetas al estilo ColdFusion, pero permite utiliza un lenguaje script, por defecto VBScript, un subconjunto de Visual Basic de Microsoft. Puede combinar código HTML, scripts y componentes ActiveX del servidor para crear soluciones dinámicas y muy potentes para la Web. Microsoft introdujo ésta tecnología como parte del Internet Information Server (IIS). El script por dfecto es el VBScript, pero existe otra diversidad de lenguajes que pueden ser utilizados como lo es Perl, JScript, etc.

El ASP es una tecnología dinámica funcionando del lado del servidor, lo que significa que cuando el usuario solicita un documento ASP, las instrucciones de programación dentro del script son ejecutadas para enviar al navegador únicamente el código HTML resultante.

La ventaja principal de las tecnologías dependientes del servidor radica en la seguridad que tiene el programador sobre su código, ya que éste se encuentra únicamente en los archivos del servidor que al ser solicitado a través del web, es ejecutado, por lo que los usuario no tienen acceso más que a la página resultante en su navegador.

Sus funciones principales están el acceso a bases de datos, envío de correo electrónico, creación dinámica de gráficos y otros. Muchas cosas que se pueden realizar por medio de CGI pueden ser realizadas con esta tecnología.

Esto es debido a que el ASP es tan eficiente como escribir código directamente en la interfaz de aplicación del servidor, con la ventaja de que es más eficiente que el CGI (que depende de un compilador) ya que el ASP corre como un servicio en el servidor, tomando ventaja de la arquitectura de multitareas. El único inconveniente con las ASP es que solamente trabajan sobre plataformas Windows 9x, NT y Linux con las aplicaciones correspondientes y sus lenguajes de programación son limitados: VBScript, Perl, JScript y ActiveX.

Nuestra elección es el Windows XP, ya que se trata de uno de los sistemas más fiables de los que encuentra la familia XP y nos permitirá una fácil instalación y configuración.

Servlets

La tecnología Servlet proporciona las mismas ventajas del lenguaje Java en cuanto a portabilidad ("weite once, run anywhere") y seguridad, ya que un servlet es una clase de Java igual que cualquier otra, y por tanto tiene en ese sentido todas las características del leguaje. Esto es algo de lo que carecen los progrmas CGI, ya que hay que compilarlos para el sistema operativo del servidor y no disponen en muchos casos de técnicas de comprobación dinámica de errores en tiempo de ejecución.

Otra de las principales ventajas de los servlets con respecto a los programas CGI, es la del rendimiento y esto a pesar de que Java no es un lenguaje particularmente rápido. Mientras que es necesario cargar los programas CGI tantas veces como peticiones de servicio existan por parte de los clientes, los servlets, una vez que son llamados por primera vez, quedan activos en la memoria del servidor hasta que el programa que controla el servidor los desactiva. De esta manera se minimiza en gran medida el tiempo de respuesta.

Además, los servlets se benefician de la gran capacidad de Java para ejecutar métodos en ordenadores remotos, para conectar con bases de datos, para la seguridad en la información, etc. Se podría decir que las clases estándar de java ofrecen resueltos muchos problemas que con otros lenguajes tiene que resolver el programador.

Además de las características indicadas, los servlets tienen estas otras:

1. Son independientes del servidor utilizado y de su sistema operativo, lo que quiere decir que a pesar de estar escritos en Java, el servidor puede estar escrito en cualquier lenguaje de programación, obteniéndose exactamente el mismo resultado que si lo estuviera en Java.
2. Los servlets pueden llamar a otros servlets, e incluso a métodos concretos de otros servlets. De esta forma se puede distribuir de forma más eficiente el trabajo a realizar. Se podría tener un servlet encargado de la interacción con los clientes y que llamara a otro servlet para que a su vez se encargara de la comunicación con una base de datos. De igual forma, los servlets permiten redireccionar peticiones de servicios a otros servlets en la misma máquina o en una remota)
3. Los servlets pueden obtener fácilmente información acerca del cliente (la permitida por el protocolo http), tal como su dirección IP, el puerto que se utiliza en la llamada, el método utilizado (GET, POST, …), etc.
4. Permiten además la utilización de cookies y sesiones, de forma que se puede guardar información específica acerca de un usuario determinado, personalizado de esta forma la interacción cliente-servidor. Una clara aplicación es mantener la sesión con un cliente.
5. Los servlets pueden actuar como enlace entre el cliente y una o varias bases de datos en arquitecturas cliente-servidor de tres capas.

6. Pueden realizar tareas de proxy para un applet. Debido a las restricciones de seguridad, un applet no puede acceder directamente por ejemplo a un servidor de datos localizado en cualquier máquina remota, pero el servlet sí puede hacerlo desde su parte.
7. Al igual que los programas CGI, los servlets permiten la generación dinámica de código HTML dentro de una propia página HTML. Así, pueden emplearse servlets para la creación de contadores, banners, etc.

JavaServer Pages

De lo escrito hasta ahora se desprende que la incorporación de contenido en dinámico en un sitio Web siempre lleva consigo algún tipo de programación para indicar cómo debe generarse ese contenido dinámico. Sin embargo, la programación requiere unos conocimientos que no todo el mundo posee, tiende a resultar cara y difícil de mantener, por lo que una de las metas en la creación de contenido dinámico es minimizar la necesidad de programación y, en último caso, separar la programación de la presentación del contenido. Combinando estos objetivos, el uso de Java y la utilización de etiquetas, la tecnología JavaServer Pages (JSP) es el resultado creado por Sun Microsystems.

Aunque los servlets pueden ser un programa completo para la generación de respuesta atendiendo a peticiones del cliente, la tarea de la generación de contenido dinámico debe ser separada en dos partes, para facilitar la programación y reducir en lo posible el coste de creación y mantenimiento. Las dos partes que intervienen en la generación de contenidos dinámicos son entonces:

- Lógica de negocio, creación de contenidos, que controla la relación entre la entrada, los algoritmos y la salida.
- Lógica de presentación, presentación de contenidos o diseño gráfico, que determina la forma en que se va a presentar la información al usuario.

En este escenario, la lógica de negocio puede ser controlada desde JavaBean y la lógica de presentación puede ser manejada a través de la tecnología JSP, mientras que los servlets se encargan del control del protocolo http.

La tecnología JSP es un híbrido, porque por un lado soporta el código embebido en sus páginas, al igual que ASP, PHP o SSJS; pero por otro lado, también permite el uso de etiquetas que interactúan con objetos Java en el servidor, al igual que ColdFusion.

Con este modelo híbrido, la tecnología JSP proporciona muchas ventajas. Los desarrolladores pueden ofrecer etiquetas personalizadas que los diseñadores de páginas pueden utilizar mediante sintaxis semejante alas etiquetas HTML que ya conocen. Como el motor JSP es capaza de compilar la página JSP bajo demanda, el autor de la página puede realizar actualizaciones fácilmente. Las páginas JSP pueden proporcionar acceso a componentes JavaBeans que encapsulan la lógica de negocio, o programación, acceso a datos, etc. Estos componentes, una vez escritos, son portables entre plataformas y servidores. La reutilización de los componentes ya existentes acelera el desarrollo de nuevas aplicaciones.

Los diseñadores de páginas web pueden modificar y editar la parte estática de la página tantas veces como deseen, si afectar a la lógica de la aplicación. Del mismo modo que los desarrolladores pueden introducir cambios en los algoritmos a nivel de un componente JavaBean sin tener que editar cada una de las páginas que utilice ese componenete.

Bases de Datos

Existe una gran oferta de bases de datos en el mercado. Variando las características entre ellas, estando cada una orientada hacia una determinada labor y soportando una determinada carga de trabajo.

Algunos ejemplos de bases de datos que existen son Oracle, SyBase, MS SQL Server, InterBase, MySQL, PostgreSQL, DBase, Access... Entre ellas haremos una breve descripción sobre sus características para poder después decidirnos sobre alguna de ellas.

De entre todas, probablemente la más conocida de todas es Oracle. Esta se considera como la más robustas y segura del mercado. Posee una gran capacidad a la hora de hacer transacciones, administración de contenido, o el llamado bussiness intelligence. El problema que surge es que Oracle viene a ser una base de datos comercial, orientada a las grandes empresas, que soportan un elevadísimo número de usuarios así como de transacciones, procesamiento de datos, etc lo que hace que las licencias software de esta compañía no estén al alcance de muchos usuarios de a pie.

Otra base de datos de uso muy extendido es MySQL. Su éxito reside en que MySQL no depende de la plataforma, existiendo versiones tanto para sistemas operativos GNU/Linux como para Windows. Otra de las ventajas que posee es que es gratuita y soporta múltiples accesos simultáneos (aunque sin llegar a los grandes volúmenes soportados por bases de datos como Oracle). Otras características es que es rápida, potente y precisa.

MS Access es la base de datos de Microsoft y que podemos encontrar en el paquete software MS Office. Esta base de datos es de muy fácil e intuitiva utilización en su entorno gráfico. De todas formas tiene dos grandes desventajas, es dependiente de la plataforma (sólo válida para Windows, además de acarrear el pago de la respectiva licencia), no soporta muy bien la simultaneidad de un número elevado de usuarios.

Por último la base de datos de la compañía Borland, InterBase, ofrece unos muy buenos resultados en varios aspectos. Existe una versión gratuita (open source) eliminando las trabas que suponen las licencias, admite un elevado número de accesos de usuarios sin degradar en demasía su rendimiento y por otro lado no es en absoluto complicada de manejar. Otro punto a su favor es que existen diferentes versiones open source para GNU/Linux y Windows.

4.3.4 La elección

En primer lugar nos hemos decantado por utilizar como lenguaje de programación JSP y Servlets ya que presenta una serie de ventajas ya comentadas respecto al resto. Además esta opción es la más adecuada para separar la interfaz de la implementación.

La elección de JSP y Servlets nos facilita la elección del servidor y sistema operativo a elegir, ya que estos lenguajes se caracterizan por su flexibilidad que le permite funcionar en varias plataformas.

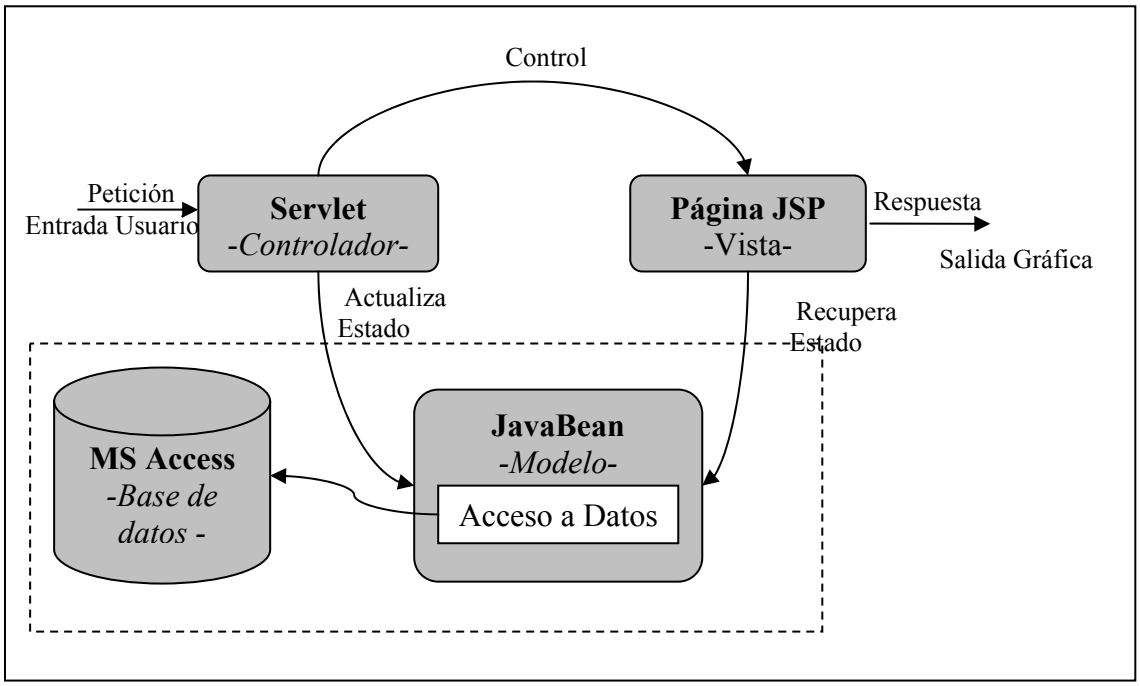
Una vez que tenemos libertad para elegir un sistema operativo, nos decantamos por el Windows XP, ya que aporta seguridad, una aceptable estabilidad y sobre todos nos permitirá la correcta instalación y configuración gracias a su interfaz amigable y fácilmente usable.

En el servidor nos decantamos claramente por Apache y su módulo Tomcat ya que son los más adecuados para interpretar y ejecutar código servlet y JSP, además está presente en la plataforma elegida, Windows XP.

La elección de la base de datos ha sido la más delicada, si bien MySql e interBase son unas bases de datos muy potentes, flexibles y gratuitas, nos decantamos por MS Access, ya que aun no siendo tan potente, sí es en cambio muy sencilla de utilizar. Además es a la vez la más común entre los ordenadores de los usuarios a los que va dirigida esta herramienta. Por este motivo nos decantamos por esta opcción.

4.4 Modelo final utilizado

Como ya se ha comentado, nos hemos decantado por una arquitectura en n-capas con el patrón MVC utilizando servlets, JSP y MS Access. A continuación mostramos una figura en la reflejamos nuestra elección.



4.5 Elección de la Metodologías

Siempre que se desarrolla una pequeña aplicación, se suele pensar que no es necesario realizar una fase de análisis y otra posterior de diseño, ya que se piensa que es una pérdida de tiempo y de recursos.

Esto es totalmente falso, ya que es necesario seguir una metodología de análisis y diseño para posteriormente implementar la aplicación, pues son muchas las ventajas que esto conlleva incluso para los pequeños proyectos.

Si para un proyecto pequeño ya es vital la utilización de una metodología, no hablemos del caso de una aplicación más compleja e importante. Pues la metodología nos permitirá hacer frente a la coordinación de todas las personas que forman parte del equipo de desarrollo, lo que repercutirá en una utilización más optima de recursos humanos, materiales y temporales. Lo que a su vez nos permitirá ajustar los costes de desarrollo.

Como podemos ver, la utilización de una metodología de diseño del software es completamente necesaria y vital para el buen desarrollo de nuestra aplicación. Lo siguiente que hay tener en cuenta es que es preciso que la notación que se utilice sea conocida por todos los integrantes del proyecto.

En el desarrollo de esta aplicación emplearemos una metodología orientada a objetos, más concretamente mediante UML (Unified Modeling Language). Esta metodología divide cada proyecto en una serie de vistas que son representadas mediante diagramas y la conjunción de todos ellos representará la arquitectura del proyecto.

UML permite realizar un modelado común para todos los desarrollos y por tanto la documentación también lo es. Lo que permitirá que cualquier desarrollador con conocimientos en UML será capaz de comprender e interpretar el desarrollo utilizado, independientemente del leguaje que posteriormente utilice para su implementación.

UML es un estándar no existiendo otras especificaciones de diseño orientado a objetos tan potente y útiles.

4.6 Bibliografía

[1] JavaServer Pages – Manual de usuario y tutorial
Agustín Froufe
Ra-Ma

[2] Aprenda Microsoft Access 97 como si estuviera en primero
José Maria Sarriegui, Nicolás Serrano e Iván Muro
Escuela Superior de Ingenieros Industriales
Universidad de Navarra

[3] Estrategias en la construcción de servidores de información
Cuaderno del curso de extensión Universitaria de Oviedo
8 de abril al 10 de mayo de 2002

[4] Aprenda Servlets de Java como si estuviera en segundo
Javier García de jalón, José Ignacio Rodríguez y Aitor Imaz
Escuela Superior de Ingenieros Industriales
Universidad de Navarra